

## The Delivery of Enabling Software as a Service

**Contacts:**

Quocirca Ltd  
Tel +44 118 948 3360  
[Clive.longbottom@quocirca.com](mailto:Clive.longbottom@quocirca.com)

The downturn that followed the collapse of the dot-com has resulted in an industry-wide drive towards the delivery of Information Technology (IT) in a way that best delivers business value. This initiative goes under a number of names, including Microsoft's "Agile Enterprise" as well as "Business On Demand" and the "Adaptive Infrastructure". These instantiations do not represent any one technology, but they all share a vision of how IT can be delivered in some way as a *service* to the business – a service where we can call on functionality to enable dynamic business processes, rather than being dependent on large, monolithic applications and dedicated servers to manage complete areas of the business.

The purpose of the report is to consider one layer of IT – the Enabling Software layer – in the context of this vision. This layer consists of a number of software services covering functionality such as databases, content management, transaction and presentation management, and a variety of other software. These packages are considered as "enabling software" as they don't deliver applications in themselves, rather they incorporate a number of functions that can be used by applications and/or other calling functions. If we wish to deliver applications as a service, the enabling software layer also needs to be delivered as a service.

**REPORT NOTE:**

This report has been written independently by Quocirca Ltd to address certain issues found in today's organisations. The report draws on Quocirca's extensive knowledge of the technology and business arenas, and provides advice on the approach that organisations should take to create a more effective and efficient environment for future growth.

While there are a number of business-oriented benefits to this service model, there are also a number of issues and concerns that are preventing organisations from adopting it. All the same, there is sufficient evidence that the delivery of software as a service is a positive step to take. By looking at these areas of weakness, we can propose an approach to overcome them.

This report treats these topics as a whole. The benefits are presented here, back to back with the issues, and a set of principles is given to help maximise an organisation's potential for success. The goal is to offer a coherent set of starting points for any organisation, with evolution, not revolution, being the key. As such, this report presents a series of evolutionary stages to help organisations understand where they are and what steps they can take to move forward.

Deliberately non-technical, this report can be read by anyone who wishes to have a broader understanding of how software can work better to deliver applications that meet the needs of the business. It incorporates examples of products from Microsoft, but its advice is appropriate for users of all environments.

*An independent report by Quocirca Ltd.*

[www.quocirca.com](http://www.quocirca.com)

Commissioned by Microsoft Inc.

**Microsoft**

**quocirca**

# 1 Contents

1	Contents.....	2
2	What is Enabling Software?.....	3
3	Benefits of Software as a Service.....	5
4	Barriers to Software as a Service .....	6
5	Overcoming the Barriers .....	8
5.1	Start from Business Requirements.....	9
5.2	Make Business Value the Central Driver.....	9
5.3	Use Business Policy to Drive SLAs .....	10
5.4	Deploy a Best of Breed Solution.....	10
5.5	Enable Self-service to the Service User.....	11
5.6	Incorporate Mechanisms for Migration .....	11
5.7	Architect for Service Delivery .....	11
6	Evolving Towards a Service .....	12
7	Conclusion .....	13
8	Appendix A. Service Oriented Architecture .....	14
	About Quocirca.....	15

## 2 What is Enabling Software?

Business users have historically required business applications so that they can carry out their daily work. There is a good deal of common functionality in these business applications however, and it is both problematic and pointless to duplicate this functionality in each application. Instead, many of today's applications depend on a layer of enabling software which enables them to share areas of functionality. This layer does not deliver applications directly; it exists to serve the needs of applications, to manage the common services that these applications require. To provide these services, the enabling software layer incorporates a large catalogue of software components. The categories are broad – some exist only because of tradition, and many have overlapping functionality. They include, but are not limited to:

- **Database Management**

Databases support the management of structured content, traditionally information that is appropriate to be stored as columns and rows of database tables. Examples here include areas such as customer data, inventory, financial transaction data and so on. Databases incorporate mechanisms for handling the simultaneous manipulation of data sets by multiple users.

- **Content Management**

Content management refers to the organisation and use of unstructured content: that is, documents and files, including web pages. Much of a company's intellectual property is held in this unstructured manner – and the rise of governance and audit is driving companies to gain greater control over these assets. Content management has extensive overlaps with software configuration management (managing the lifecycle of software development through changes, revisions and implementation/roll back), and many content/configuration management tools perform both functions.

- **Transaction Management**

Transaction servers have existed for almost as long as transactions; a transaction may be seen as the pairing of a service request with a service response. An e-Commerce payment is a transaction for example, as is a flight booking or a database update.

- **Directory Management**

A computerised directory is a form of specialised database, designed to store information about everything computer systems need to know in order to deliver their services. This can include user details, application configurations, system assets, software functions and so on.

- **Process Management and Workflow**

This software enables users to organise their work according to sequences of activities. These are frequently and inevitably the same activities that make up business processes, discussed later in this report. Other applications can benefit from explicit workflow capabilities, for example document production and authorisation.

- **Presentation Management**

Standardised user interfaces enable software applications to be accessed in a common way. The most common interface for applications is as software running on the desktop or a terminal services client, as this has historically allowed for the richest user experience. Increasingly, many applications deliver a slightly reduced functionality set via a Web

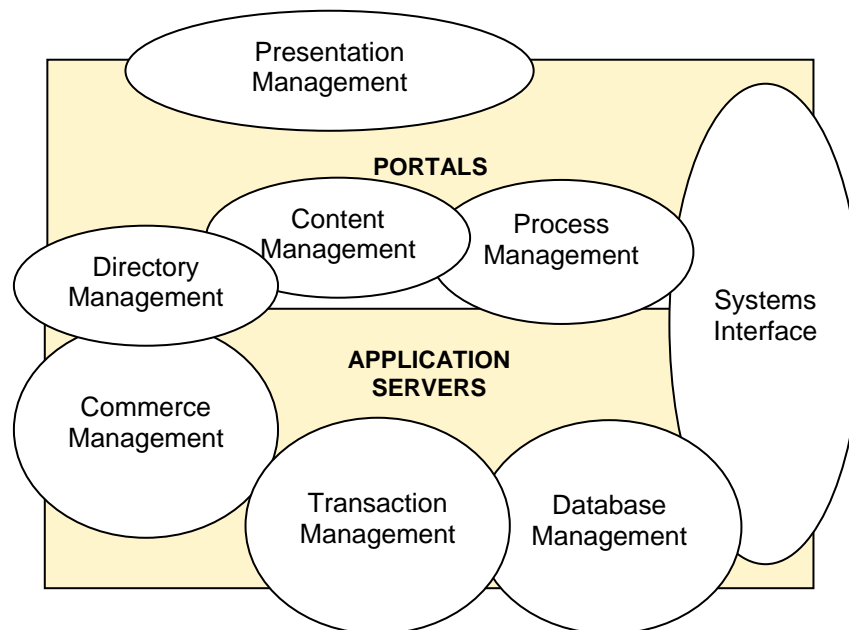
Browser interface, which can allow an application to be accessed wherever there is Internet access.

- **Commerce Management**

Web commerce functionality arrived simultaneously with the e-Commerce boom of the late 1990s. It incorporates such capabilities as shopping baskets, product catalogue management and inventory search, and is often dependent on other back end services such as transaction management, workflow and so on.

Given such a list, the biggest issue is the need for integration of the various services required to fulfil the process: ensuring that the different pieces of the puzzle work together smoothly to solve our business need. There is also the need for interfaces between existing applications and the new services. For example, one application may need to access data from another in-house application, or from an application owned by another company. These interfaces can often be implemented with Web services, as part of a Service Oriented Architecture (SOA – see Appendix 1). Asynchronous (non-real time) interfaces may be required when a response to a request cannot be guaranteed within a certain time frame. While the official definition considers asynchronous messaging products such as Microsoft Message Queue, E-mail can also be considered as a mechanism for asynchronous communications.

Some products, such as portal and application server software shown in the diagram below, already integrate several (but not all) of these components. Microsoft has taken things one step further, and incorporated a number of these elements directly into the Windows operating system – notably transaction, directory and presentation management functionality.



All of the product categories given above share the same goal – the management of business information and support to the applications that use it. To enable business applications to deliver a service however, each category and the integrated result needs also to deliver a service. As well as meeting functionality requirements, this means satisfying the following set of service criteria:

- **Consistency** – The well-established SLA-type qualities of availability, scalability and so on yield the requirement to provide a consistent service, according to a set of characteristics agreed with the business.

- **Adaptability** – A service should be self-maintaining and self-optimising as much as possible, in other words it should be able to maximise its own value. This often equates to automating management tasks.
- **Provisioning** – The ability to switch a service on and off forms the basis for the ability to provision a service. This goes back to back with accountability (below), and is also the basis of ensuring applications are delivered securely.
- **Accountability** – This refers to allocation of the costs of a service to the most appropriate cost center. While this incorporates billing and cross charging, in many environments it is as important to demonstrate and allocate costs even when no billing takes place. Together with security, accountability is an enabler to corporate governance.
- **Security** – The need to deliver a secure service should need no justification! While security management software could be seen as a separate package of functions, the reality is that every function needs to be delivered securely – and that security has therefore to be implemented as a central service across the complete system.
- **Accessibility** – a service needs to be accessed by the service customer, in a way that makes sense to the customer. At the lower levels this equates to standards; for human-facing services, it equates to usability.
- **Management** – suitable interfaces need to exist to ensure the ongoing operation and management of the service. This includes the delivery of metrics, which can provide an understanding of how well the service is being delivered.

There is clearly a lot to be done to deliver the enabling software layer in an integrated fashion, supporting these seven criteria. If this was achieved however, what benefits might businesses expect?

### 3 Benefits of Software as a Service

The delivery of information technology as a service to the business offers some obvious benefits, based on the fact that less IT can be used to greater effect. For the purposes of this report, it is interesting to consider the benefits that can be directly related to the provision of a comprehensive, integrated enabling software layer as a service. These include:

- **Reduced deployment lead times**

The existence of an integrated, enabling software layer means that less time needs to be spent re-inventing the wheel. This leaves more time for the development of innovative application functions. As shall be seen below the sooner an application can be delivered, the sooner it can start to deliver business value – an area known as “Time to Capability”. For customer-facing applications in a competitive environment, application time to capability can have a direct impact on the top line of the business.

In addition, businesses can work faster and focus more on their core activities, rather than wasting time on unnecessary software development.

- **More cost effective functionality**

With the best will in the world, there are some functions that will never be cost-effective to develop for a single organisation, and to a large extent these already exist elsewhere. The enabling software layer will always deliver more enhanced functionality, with a wider set of interfaces, than can be provided if built from scratch.

- **Reduced margin for error**

Software inherently contains errors. A software package that has been procured and that has already gone through multiple iterations is likely to come with fewer errors out of the box than if the same package is built in-house. This may sound controversial, as we can all think of examples of bugs in vendor software or ultra-reliable legacy systems, but in most cases software bought off-the-shelf will have been used far more rigorously, in more organisations, than software built in-house.

- **Adherence to standards**

Use of third party enabling software makes it easier to adopt industry standards, as and when they come into force. This applies both to technical standards such as the eXtensible Markup Language (XML), transactions standards such as EDI (Electronic Data Interchange) and business standards such as accounting best practice. Once again, it makes more sense for a vendor to solve the problem once, than for an end user to solve the problem many times.

- **Reduced operational costs**

The delivery of Enabling Software as an integrated structure should have a positive impact on the costs of its monitoring, deployment and maintenance. This is due not only to integration issues inherent in deployment and upgrade, but also enhanced manageability when multiple packages are controlled from the same point, rather than having to rely on multiple management interfaces. There is also less hardware required for the same functionality, as less compute power is required to run a function once rather than across multiple different applications.

Clearly these benefits are worth having, but for many organisations they have come at too great a cost to be feasible. It is worth taking a long, hard look at the issues that are faced: at the very least, we need to know – if this is all common sense, why is it currently so difficult and why has it taken so *long*? Perhaps more importantly, why should things work now or in the future, if they haven't before?

## 4 Barriers to Software as a Service

It is all very well putting together a blueprint of perfection, but the reality is not as simple. There are some valid reasons why we are still trying to achieve the goals of software as a service. As you can see from the list below, many of the issues are little to do with the technology itself. Between vendor, service provider, consultant and end-user organisation, each has played their part.

- **Lack of integration**

While this report suggests the desirability of having an integrated enabling software layer, most organisations have anything but. Integration is an issue largely because it wasn't considered at the outset – silo-based software packages just weren't designed to communicate with each other, and it has proven difficult to retrofit such a capability.

Various integration techniques have been employed in the past, and a number of products deliver some kind of middleware or Enterprise Application Integration (EAI) capability. In today's environments, there exist a plethora of point products which provide interfaces between legacy applications. However, each interface increases the latency of any transaction; the resulting mass of products can be less than stable, making it difficult to remove or replace one product for the fear of bringing down the ensemble.

Until recently, some technologies have just not been ready for delivering an integrated service. With the arrival of the standards efforts that support SOA, it looks like this situation is beginning to change. Standardised interfaces enable IT vendors to build software packages that work together from the outset, rather than requiring proprietary technologies with an inevitable performance impact.

- **Lack of business process focus**

Until recently, the direct linkage between IT and business activities has been seen by many as a nice-to-have, rather than an essential element of delivering IT as a service. In many cases, workflow software has remained no more than a bolt-on to content management tools. This is despite the repeated discovery that integration middleware can only function properly if primed with an understanding of business processes.

Fortunately, workflow and process management tools are becoming more commonplace and more accessible. In Microsoft's toolset for example, workflow and orchestration tools are built into Biztalk Server 2004, Visual Studio 2005 and SharePoint Server.

The next step of course, is to maximise companies' use of such tools. In the past, the focus has often been on automating people *out* of the process, but there are plenty of opportunities to consider how to automate people *in*, enabling human intervention and maximising the opportunity for personal service – but in context, maximising the cost-effective use of these interactions.

- **No time for businesses to stop**

Businesses do not operate in a vacuum, and cannot close down for a fortnight when they want to implement the latest software package. Deployments have to cause minimal disruption; otherwise they do not happen at all. In the past this has often led to more tactical deployment of software, which is one element of why existing IT environments are fragmented.

The emergence of more comprehensive platforms of enabling software, makes it easier for companies to adopt pre-integrated technologies with reduced risks and deployment timescales. For this reason also, the only sensible path is an evolutionary path. Revolution was never, and is still not, an option.

- **Businesses washing their hands of IT**

Just as IT departments do not always fully cross the chasm of understanding the business they are in, so businesses have sometimes been reluctant to take responsibility for their use of IT. Quite rightly, businesses want IT to just work. Indeed, so it should, but it works best when it knows what it is for.

It was not really until the rise of the dot-com that many businesses started to appreciate the value that IT could bring, if they only spent some time to work out how best to use it. Businesses are far better at this than in the past, taking a more hands-on approach to both requirements definition and technology deployment, but there is still room for progress. Understanding the business imperative, decomposing this to the business processes and tasks, and matching these against the technical capabilities required will help in maximising the flexibility and capability of a service-based approach.

- **Investment Protection**

If companies did everything they were told by IT vendors and industry advisors, they would throw everything away and start again with whatever is being promoted. Fortunately they do not, and history tells us that technology replacement is not always the best option. For example, EDI was a vast investment that is currently working very well, largely supported by external service providers that are delivering a scalable

infrastructure for EDI transactions. Companies are not just going to replace EDI by what is perceived as its Internet-based poor brother, however pervasive the Internet might be. Nor should they have to – in this case, EDI can be seen as a perfectly valid piece of the puzzle.

Integration standards are a solution to investment protection. In this particular case, for example, XML to EDI integration can be supported using Microsoft BizTalk Server. Protecting an investment will not always be the best approach, particularly if increased benefits can be achieved from a new technology, but businesses should be able to make such a decision for business reasons and not because they are forced to.

- **Data complexity**

It has been said that the utilities market is self-regulating. Indeed there is anecdotal evidence to suggest that, as utility companies have so little control over the quality of their customer data, it is impossible for them to do anything that might be in breach of the customer's best interests. Time and again, there are examples of how poor data quality causes many difficulties, in migration to new systems or upgrading old ones. This manifests itself in a number of ways, not least the issue of data synchronisation across organisations in a supply chain.

Data complexity is one of the greatest unsolved problems of IT. By calling each database a legacy system, a migration path can be instigated, but this is a slow solution to a major problem. As we shall see below, it is possible to build data migration plans into the delivery of software as a service.

- **Self interest**

Finally, we cannot totally ignore the fact that a number of the issues have been, to an extent, driven by self interest. No party is entirely innocent of this charge: in the past, vendors have focused on more proprietary technologies, leading to fragmentation of the market. Meanwhile, changes in IT have often been seen as a way of increasing automation and therefore, reducing head count: nobody is going to willingly put themselves out of a job.

However, with the efforts that have gone into the definition of Web services standards, the majority of vendors have agreed which hymn sheets to sing from – more or less. Standards have replaced a number of proprietary technologies, which is to be applauded. In the end user camp, it does not make business sense to preserve the status quo for its own sake.

The issues listed here have led to poor delivery of IT as a service. If you want to get there, don't start from here, goes the adage, but we have no other starting place. Fortunately, there are a number of steps organisations can take, to start moving in the right direction.

## 5 Overcoming the Barriers

Delivery of software as a service is not some imaginary ideal; neither is it yet a practical reality for many organisations. If you consider the benefits to be worth having, then you will need strategies to overcome the issues. This is as true for computer companies, integrators and service providers, as it is for end user businesses.

All organisations are different, so there can be no hard and fast doctrine. Instead, here is a series of principles that can be applied to any organisation, to help towards the goal of delivering software as a service.

## 5.1 Start from Business Requirements

The corporate IT environment exists for a single reason – to support and automate business activities. Business requirements give IT its scope – by understanding the business activities we can understand what we want to support and to automate. There are a number of ways of doing this, the most prevalent being to define it in terms of business processes. To avoid confusion we can quote Mike Hammer, who defined Business Processes back in 1985.

*“A Business Process is a series of activities that takes a number of inputs and delivers an output of value to the customer”<sup>1</sup>*

We can support an understanding of business processes with an understanding of business policies, that is, collections of business rules. Together, defined processes and policies give us a starting point for the business case that should lead to any IT decision. By starting with the business need, we can define far more tangible requirements than if we start from the perspective of a product category. “We need to deliver a more efficient sales process,” is a much more powerful statement than, “We need a content management system”.

The goal is to end with the business as well as starting with it: the resulting application should be able to report on how well IT is doing in delivering business value. By ending with the business, an organisation can discover whether or not its IT decisions were correct, and it is in therefore a better position to make decisions in the future.

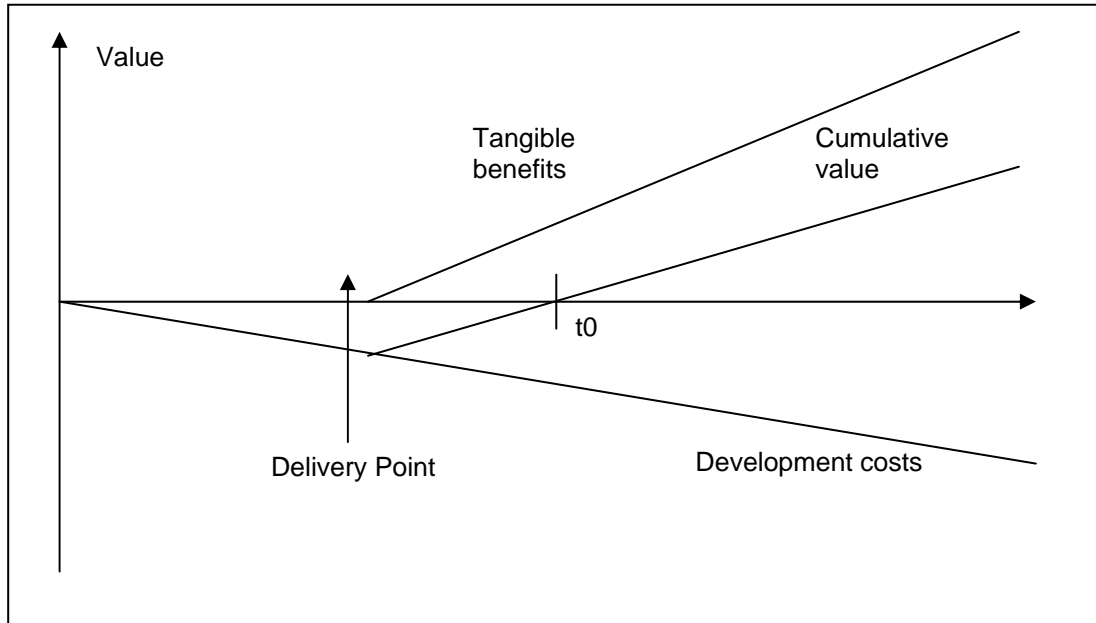
## 5.2 Make Business Value the Central Driver

There is a proviso to the Mike Hammer’s definition of a business process. Most businesses are not being altruistic – they are in the game to make money. Therefore, we need to extend our thinking to incorporate the concept of business value – the output should also be of value to the business. Business value equates to business benefits minus business costs. It must at the very least be financial, augmented by non-financial criteria.

The earlier business value can be achieved, the better. Consider, for example, the following graph, which plots value against time. When an IT project is in development, it is a net cost to the business, until such time as it has been delivered and starts to be used. Time t0 is the break-even point, where the cumulative value of any benefits has equalled the cumulative costs.

---

<sup>1</sup> “Reengineering the Corporation: Manifesto for Business Revolution”, Mike Hammer and James Champy, 1985



This picture shows the importance of early delivery of value to an organisation. As soon as a function is deployed and used successfully, a company can start to derive benefits from it; this derived value can be offset against the ongoing costs of any continued development.

### 5.3 Use Business Policy to Drive SLAs

Business policy can be used to define an organisation's expectations on system availability and performance. For example, an agreed policy for customer responsiveness dictates the length of time a business process should take, and therefore the limitations on any delays imposed by IT. Business policy also drives security policy, which has a direct IT impact.

Once a business policy is defined and the resulting applications have been implemented, you will need to monitor how well the service is being delivered. While there may be a lot of complexity within the enabling software layer, most important to the business is to know what services are available, how well they are operating, and how much they are costing.

### 5.4 Deploy a Best of Breed Solution

Currently there is no single product, or indeed single vendor offering, that covers all the necessary bases the enabling software layer should support. As a result organisations need to develop a best of breed solution, taking a set of products and using them efficiently to support the needs of their own business applications.

"Best of breed" implies using the best tool for the job in hand. For example, for certain functions it may be more appropriate to use an optimised appliance rather than using a general purpose server running a software package. Input/Output-dependent functionality can often benefit from this approach, as appliances take advantage of specialised hardware that can analyse data streams and respond far more quickly than general purpose hardware.

Note that best of breed should not be considered as an excuse for the potential morass of point products that can result. Choices must be made based on strategic need – even a tactical solution must be viewed within a longer term strategy of a coherent architecture.

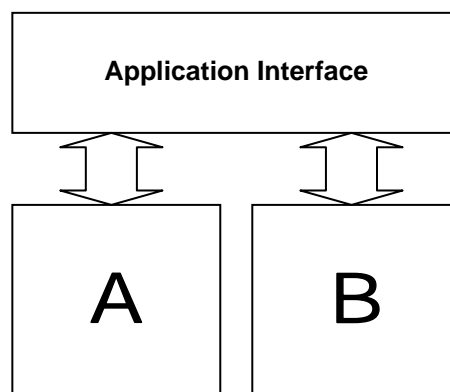
## 5.5 Enable Self-service to the Service User

It is a long established principle that a small increase in effort from the requestor of a service can significantly reduce the total cost of a business process. A trivial example is how online commerce sites require customers to fill in their own names and addresses. Not only is there an increase of data quality, there is also the potential for a greater level of data capture. Self-service techniques can be applied to IT applications in areas such as document management, resource allocation, security policy definition and helpdesk management, amongst other areas.

With self-service comes the potential for abuse. From a management perspective, however, the amount of time spent dealing with abuse as exceptions can be substantially less than the time spent dealing with normal, everyday queries as the rule.

## 5.6 Incorporate Mechanisms for Migration

It is neither desirable, nor necessary to migrate all of an organisation's applications, functionality, or data to a new software environment the moment it is delivered. It is better to deliver some kind of architecture that can support the gradual migration of data and functionality to the new environment. Consider the following diagram, which shows how migration options can exist:



In the diagram, application A is a legacy customer database, which contains the majority of past customer records. Application B is a new database, which is unpopulated. In this scenario, it is not possible to consider the migration of all data from application A to application B within a sensible timeframe. An alternative is to consider application A as a legacy data source, providing it with a data access interface. As records are used and modified, they can be read from application A and then stored to application B, offering the potential for human intervention each time.

Clearly this example does not resolve the issues of reporting and of synchronised backup. However it illustrates an alternative to a full-scale migration. Indeed, there is nothing to stop the use of data migration software to move any heavily used data from one platform to the other, before resorting to the above technique for the remainder.

## 5.7 Architect for Service Delivery

We have already said that a best of breed solution offers the best opportunity for providing the enabling software layer as a service. The service criteria given above – consistency, adaptability, security and so on need to be at the heart of the architecture.

Architectural considerations need to be at the centre of any new deployment, with all elements of the architecture working together to deliver the service criteria. Security, for

example, cannot be provided in only one part of the architecture: rather, all elements need to be appropriately secured.

Integration is the key: poorly integrated systems and software will inherently provide a worse service than well integrated ones. Exception handling for example, needs to work for every service; each service should be able to deliver its own metrics, so service provision can be monitored and managed. Infrastructure development and deployment processes also need to take an architectural view, incorporating change control and testing of the whole architecture, not just its elements.

## 6 Evolving Towards a Service

As mentioned above, it is evolution, not revolution that is key to the successful delivery of the enabling software layer as a service to its applications, and therefore, to the business. If an evolutionary approach is followed, implementation becomes a process of moving through the evolutionary stages, rather than trying to implement everything at once.

From a business perspective, the two parties involved in the evolution are the business itself, and the IT department. An organisation can progress by understanding where it is in the evolutionary scale, and then doing what is necessary to get to the next stage.

There are four evolutionary stages, as follows:

- Preparing – an organisation understands the need to work as a service, and has started to plan accordingly
- Understanding – the organisation is capturing knowledge about itself and its IT assets
- Applying – service principles are pervading the organisation's use of IT
- Optimising – the organisation uses IT as a service, and is making continuous improvement

These are presented in the table below.

	<b>BUSINESS</b>	<b>IT DEPARTMENT</b>
<b>Preparing</b>	<ul style="list-style-type: none"> <li>• Comfortable with business strategy and goals</li> <li>• Strategy and goals are communicated throughout the organisation</li> <li>• Planning roles and responsibilities to enable service based delivery of IT.</li> </ul>	<ul style="list-style-type: none"> <li>• IT organisation is defined and implemented to support IT service delivery</li> <li>• IT delivery is separate from operations</li> <li>• Management monitoring mechanisms exist</li> </ul>
<b>Understanding</b>	<ul style="list-style-type: none"> <li>• Core business processes are documented and communicated throughout the organisation</li> <li>• Interfaces with suppliers and customers are understood</li> <li>• Information requirements and states are understood</li> </ul>	<ul style="list-style-type: none"> <li>• Able to understand costs</li> <li>• Management diagnosis mechanisms exist</li> <li>• Overview of available IT assets and the services they offer</li> <li>• Risks with current IT environment defined and solutions proposed</li> <li>• Responsibilities are defined to enable service based delivery</li> </ul>

	<b>BUSINESS</b>	<b>IT DEPARTMENT</b>
<b>Applying</b>	<ul style="list-style-type: none"> <li>IT costs can be applied (or communicated) to business cost centres</li> <li>Metrics are applied to core business processes</li> <li>Linkage is made between business processes and resource provisioning processes</li> </ul>	<ul style="list-style-type: none"> <li>Management pre-emption mechanisms exist</li> <li>New purchases are considered on the basis of service</li> <li>Able to provision and charge for some applications and services</li> </ul>
<b>Optimising</b>	<ul style="list-style-type: none"> <li>Business improvements are driven through IT improvements in a cost-measured fashion, supported by live, traceable business metrics</li> <li>This is the “agile enterprise”</li> </ul>	<ul style="list-style-type: none"> <li>In-house and external IT applications and services can be costed and compared on the fly as input to a business case</li> <li>Management optimisation mechanisms exist</li> </ul>

This model is a support, not a rulebook. It is unlikely that any organisation will land squarely in one of the evolutionary stages; more likely is that certain elements will trail the other, more mature elements. Therefore, it is incumbent upon the organisation to identify where it sits most effectively at this time, and to bring any trailing elements up to speed. Once an organisation is firmly within a single area, it can then begin to improve its processes and infrastructure to evolve into the next level.

## 7 Conclusion

The central premise of this report is that organisations can benefit from evolving towards the delivery of IT as a service to the business. The enabling software layer plays an important role in this evolution, but as this implies, it can never be fully complete. Many of the tasks involved in service delivery are iterative rather than one shot operations.

Data centres are starting to be able to move away from the historical morass caused by repeated implementations of point products, towards an integrated approach that uses the best tools for the job. Once the issues of the data centre are resolved however, how well businesses succeed with IT will depend on how well they understand themselves, which is exactly as it should be.

Perhaps of all the service criteria, accountability should be seen as the most important: it has the most to offer, in terms of giving businesses a handle on their IT costs relative to the business benefits. When companies start to achieve tangible competitive advantage through better definition and application of IT, their competitors will be forced to follow suit.

## 8 Appendix A. Service Oriented Architecture

Service Oriented Architecture (SOA) is an architectural framework for writing software applications. While its principles can apply for all categories of software, it is really the enabling software layer that stands to benefit the most. SOA provides a set of principles and standards to support the development of distributed applications, as follows:

- **Modularised development.** This has gone by many names in the past, notably object orientation and component based development. Essentially, rather than considering applications as large, standalone packages of code, the goal is to define packages of functionality that are of sufficient value to exist in their own right. The object oriented approach ensures we end up with the right kinds of packages.
- **Business component reuse.** It makes sense that packages of software that are used across multiple applications need only be written once; it makes even more sense that they are not even written by companies at all, rather that they are written by software vendors. A good example of this is the Microsoft Business Framework (MBF).
- **Pre-defined execution environment.** There are also software functions that no business application software company should ever need to write, for example to control user security, managing transactions, or managing content. Today there are two environments that have gained general acceptance – .NET from Microsoft and J2EE from Sun Microsystems.
- **The Internet.** The global network that we now perceive as normal, is catalysing the need for the use of SOA. It has also driven some new requirements. For a start, there is a noticeable latency between the application request and the application response when using the Internet as a transmission mechanism. Also, it is more reliant on text-based communications.
- **A set of standards.** Standards efforts have taken place in the past (consider DCOM and CORBA/DCE), but they have never achieved universal acceptance. Today we have a combination of standards efforts, notably eXtensible Markup Language (XML) and eXtensible Stylesheet Language (XSL) for data definition and representation, and Simple Object Access Protocol (SOAP) for interfacing between application components. These standards are often described by the term “Web Services Standards”.

SOA permits the enabling software layer to act as the operating environment for distributed applications. Like most concepts in IT, the SOA has a heritage, and as such is nothing new. However, in the past it has been the practice that has proved difficult. It is the arrival of Web Services standards, and their broad adoption, that gives SOA real potential.

## About Quocirca

Quocirca is a UK based perceptual research and analysis company with a focus on the European market for information technology and communications (ITC). Its analyst team is made up of real-world practitioners with first hand experience of ITC delivery who continuously research and track the industry in the following key areas:

- Business Process Evolution and Enablement
- Enterprise Applications and Integration
- Communications, Collaboration and Mobility
- Infrastructure and IT Systems Management
- Utility Computing and Delivery of IT as a Service
- IT Delivery Channels and Practices
- IT Investment Activity, Behaviour and Planning

Quocirca research is always pragmatic, business orientated and conducted in the context of the bigger picture. ITC has the ability to transform businesses and the processes that drive them, but often fails to do so. Quocirca's mission is to help its customers improve their success rate.

Quocirca has a pro-active primary research programme, regularly polling users, purchasers and resellers of ITC products and services on the issues of the day. Over time, Quocirca has built a picture of long term investment trends, providing invaluable information for the whole of the ITC community.

Quocirca works with global and local providers of ITC products and services to help them deliver on the promise that ITC holds for business. Quocirca's clients include Morgan Stanley, Vodafone, Oracle, Ericsson, Microsoft, Orange, IBM, O2, CA and Cisco. Sponsorship of specific studies by such organisations allows much of Quocirca's research to be placed into the public domain. Quocirca's independent culture and the real-world experience of Quocirca's analysts, however, ensures that our research and analysis is always objective, accurate, actionable and challenging.

Most Quocirca research reports are available free of charge and may be requested from [www.quocirca.com](http://www.quocirca.com). To sign up to receive new reports automatically as and when they are published, please register at [www.quocirca.com/report\\_signup.htm](http://www.quocirca.com/report_signup.htm).

**Contact:**

Quocirca Ltd  
Mountbatten House  
Fairacres  
Windsor  
Berkshire  
SL4 4LE  
United Kingdom  
Tel +44 1753 754 838

Email [info@quocirca.com](mailto:info@quocirca.com)

